【完全理解】

TCPアルゴリズム

基本動作編《ロスベース方式》

- 確実にデータが届く仕組み
- ネットワーク輻輳を把握し効率的に伝送

全体目次

１．　TCP概要

1-1.　RFCとは？

1-2.　TCP/IPとは？

1-3.　プロトコルスタック

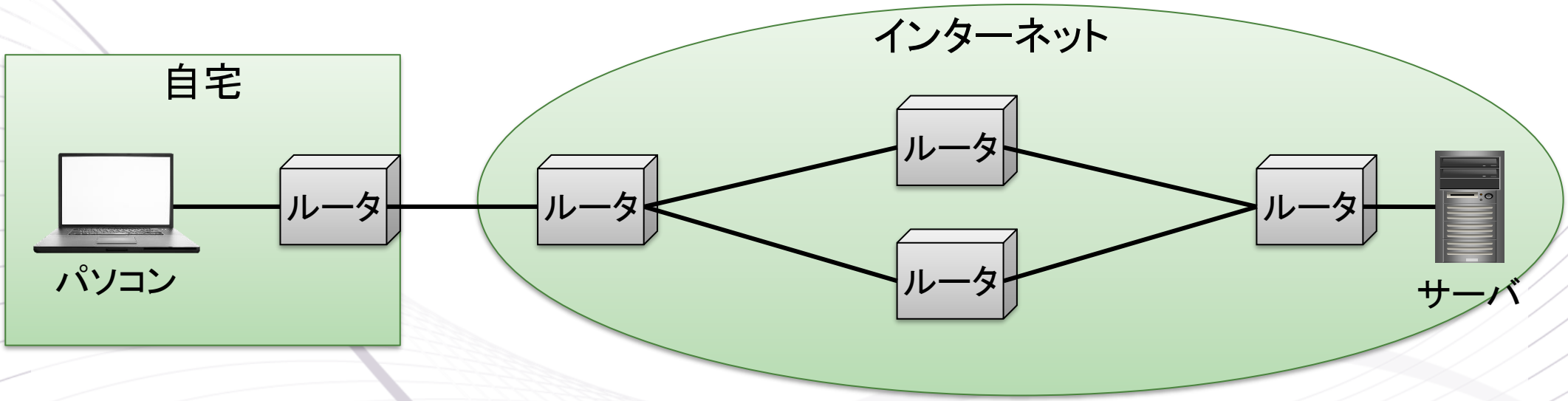
1-4.　パケットデータ（キャプチャ）

1-5.　TCPバージョン

1-6.　TCPデータフォーマット

1-1.　RFCとは？

インターネット通信に関わる全ての機器は予め規定された技術で
データを扱うことが必要。
そのため、インターネット技術の標準化（取り決め）が行われている。
この標準化をRFCという。



- **標準化を行う団体：IETF（Internet Engineering Task Force）**
- **標準化された規定：RFC（Request for Comments）**
    - **1) IP　：　RFC791**
    - **2) TCP：　RFC9293　2022年8月公開（RFC793廃止）**

1-2.　TCP/IPとは？

- TCP/IP　インターネットで通信を行うための規定

  ①IP　　：インターネット上の住所を表す役割

  ②TCP ：通信路の状態に応じて効率的かつ確実に
  　　データを届ける役割

- UDP/IP

  ③UDP ：届けるだけで通信制御をおこなわない。

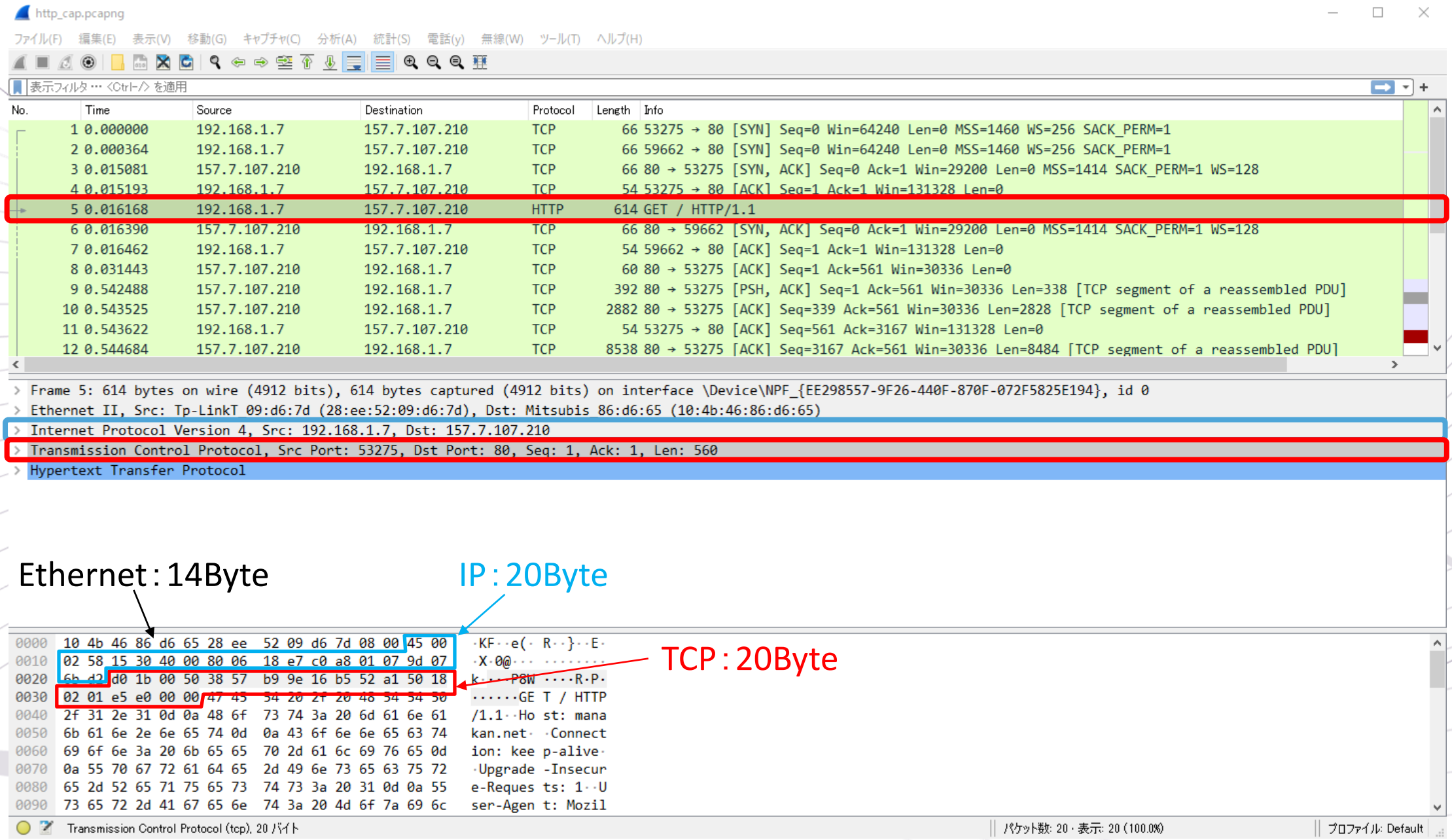
1-3.　プロトコルスタック

「DARPAモデル」

- 送受信端末で各層の利用する規定を一致させる
- 各層間の依存関係がない。
  各々の層毎に利用する規定を決める

【出典：（ビジネス＋IT） OSI参照モデルとTCP/IPの階層の違いとは？】
https://www.sbbit.jp/article/cont1/12099

1-4. パケットキャプチャ（WireShark）



1-5.　TCPバージョン

- 初期標準の最もベーシックなTCP
  - 「Reno」「NewReno」*1

今回の説明

- 高遅延・広帯域（ロングファットパイプ）向け
  - 「BIC」「CUBIC」「H-TCP」*1、「Fast TCP」*2、「Illinois」*3

- 低遅延・広帯域であるデータセンターに向け
  - 「DCTCP」*3

- モバイル回線やWi-Fiなど無線環境向け
  - 「Veno」「Westwood」*2

《参考１：方式》
*1：ロスベース方式
*2：遅延ベース方式
*3：上記２つのハイブリッド方式

《参考２：各OS実装》
Linux、Android：CUBIC
OS X：NewReno
Windows：CTCP/DCTCP

1-6. TCPデータフォーマット

- **規定　：　RFC 9293**

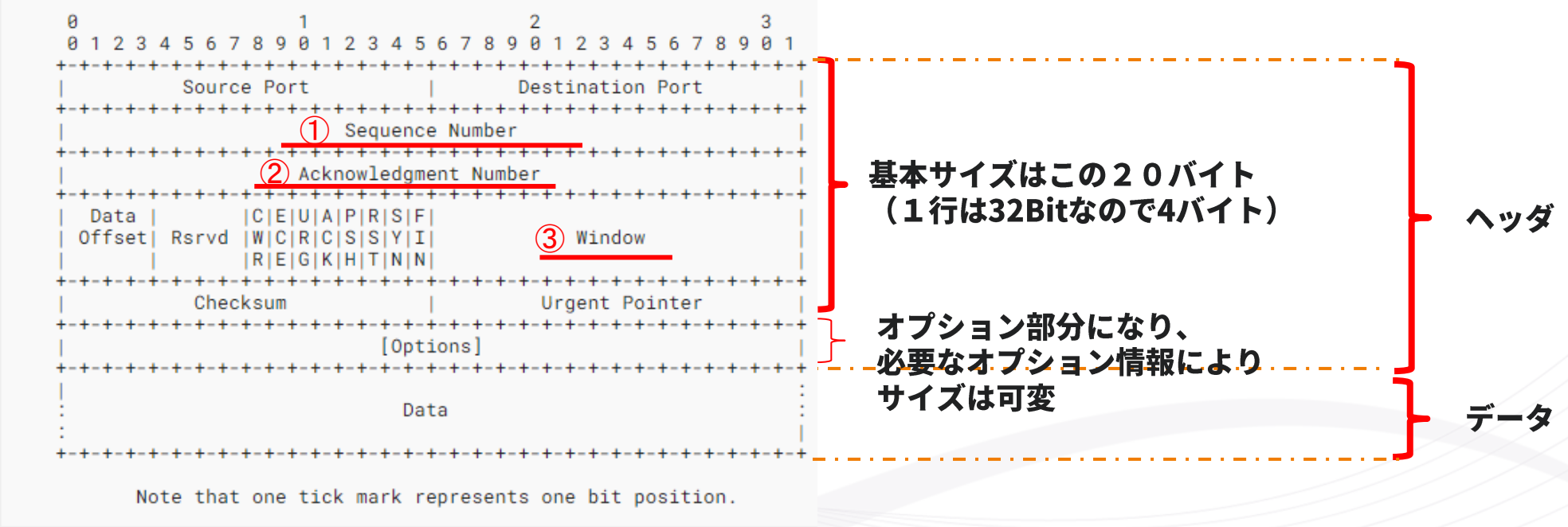


Figure 1: TCP Header Format

①	Sequence Number （シーケンスナンバー）	送信元端末が送信データに番号（Sequence Number）をつけて送信します。
②	Acknowledgment Number （アックノレッジナンバー）	受信端末が届いた「Sequence Number」を元に次に受信を期待する「Sequence Number」を確認応答（Acknowledgment Number）として送信元に通知します。
③	Window （ウィンドウ）	受信端末が受信可能なデータ量（受信バッファサイズ）を送信元に通知します。

2. TCPアルゴリズム

2-1. TCPコネクションの確立

2-2. 到達確認

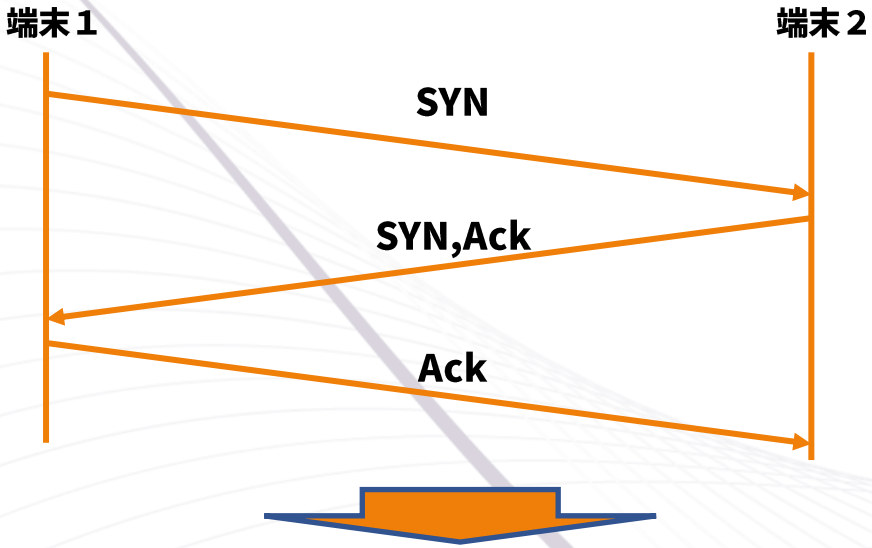
2-3. ウィンドウ制御

2-4. ウィンドウサイズの決定

2-5. 再送制御

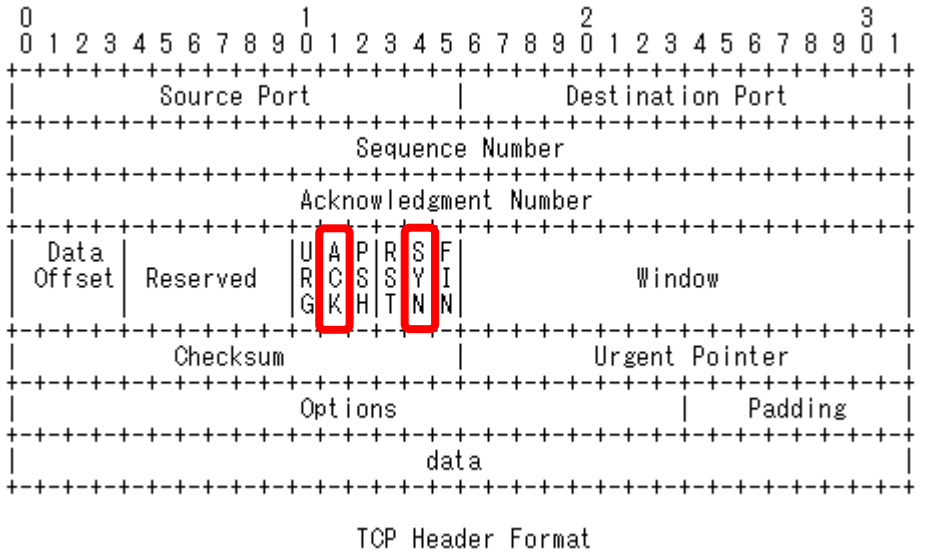
2-1.　TCPコネクションの確立

３Wayハンドシェイク



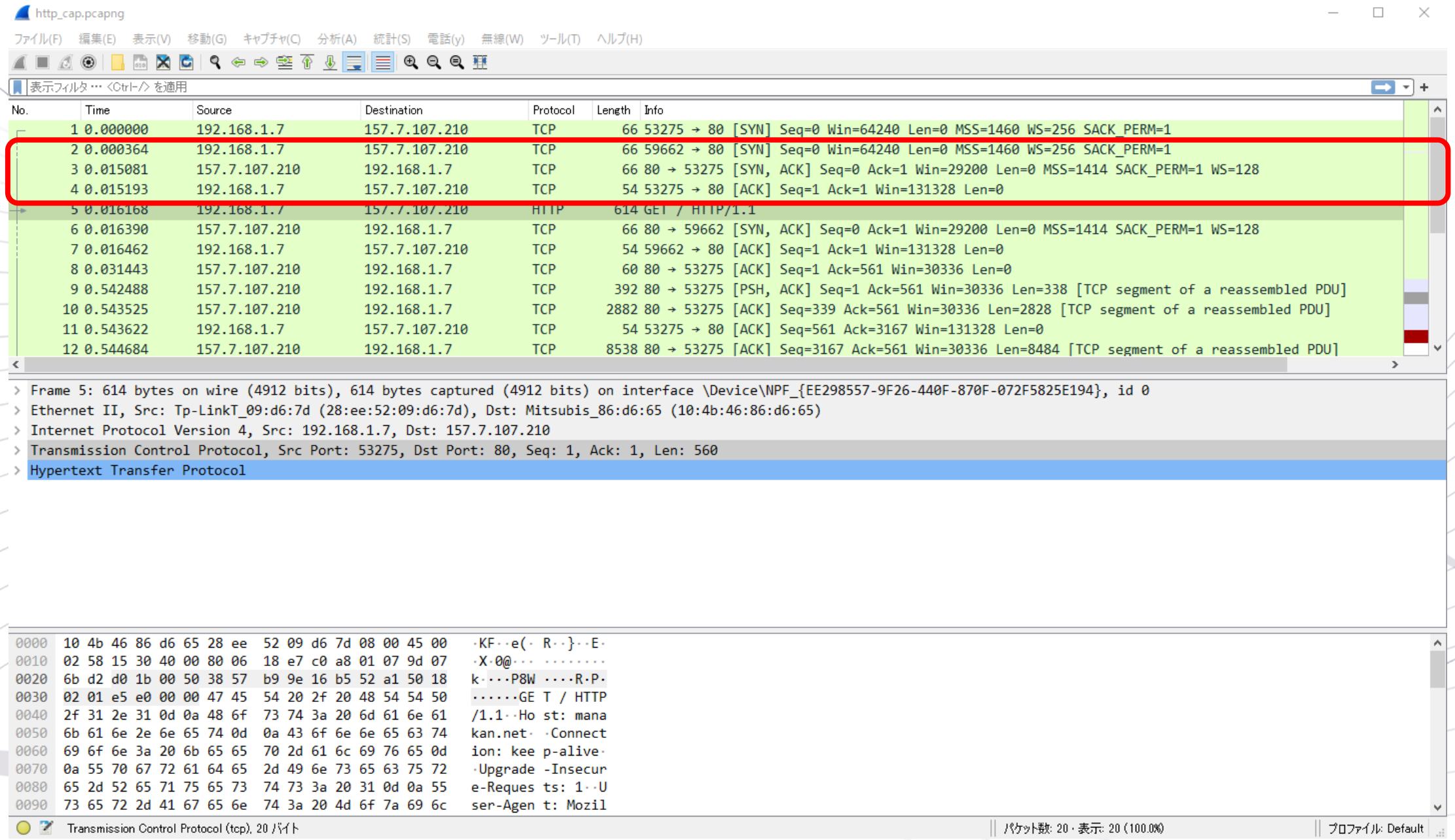
ここから以降で、データ通信が送信される

SelectiveAckやウィンドウサイズオプションなどのオプション機能の使用可否もこの確立時に決定されます。

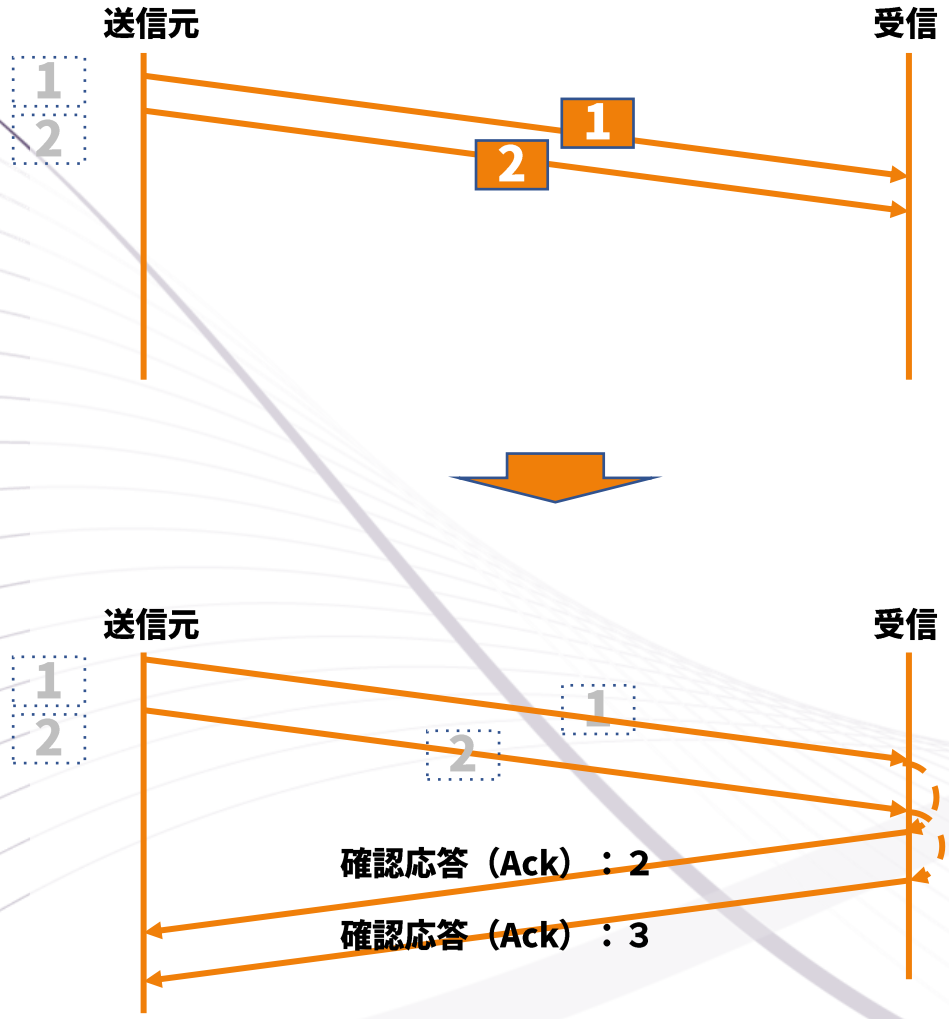


TCP Header Format

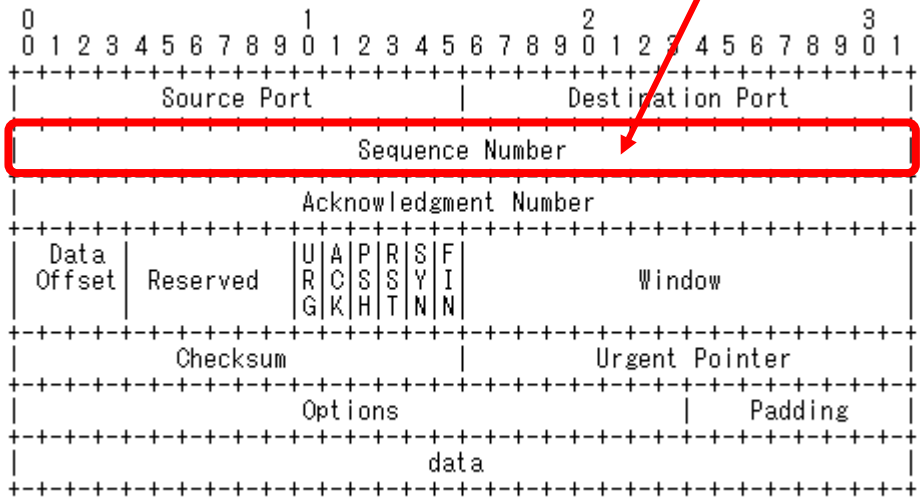
パケットキャプチャ（WireShark）



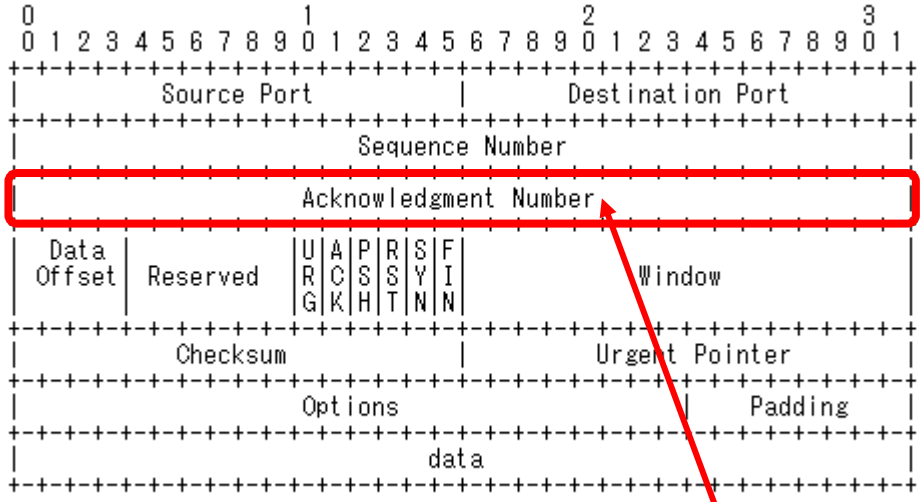
2-2.　到達確認



送信データ番号



TCP Header Format



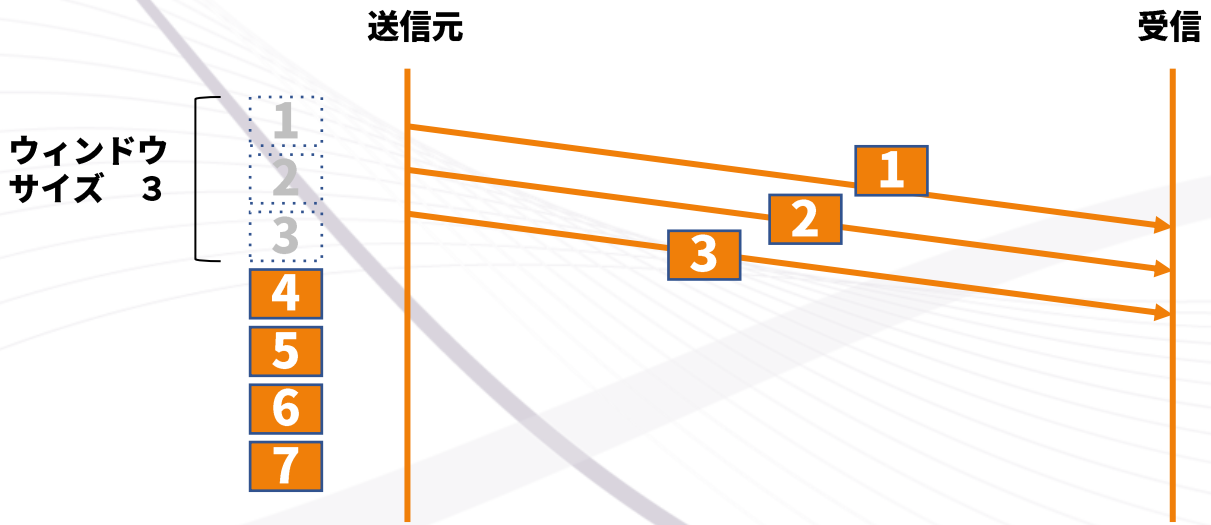
TCP Header Format

到達確認番号
（次に受ける番号）

2-3. ウィンドウ制御

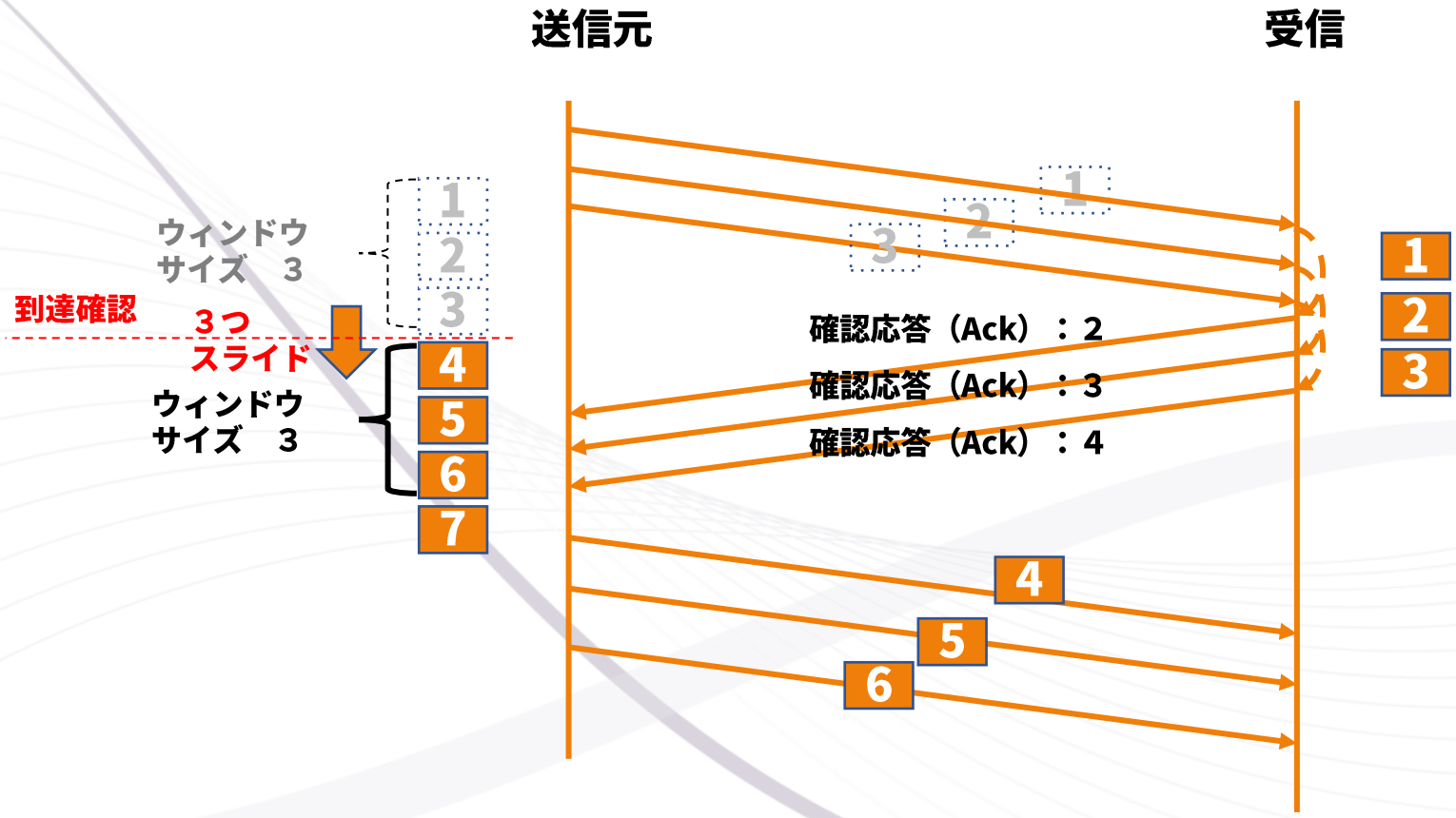
ウィンドウとは

「到達確認できていないが
送信することができるデータ量（バイト量）のこと」



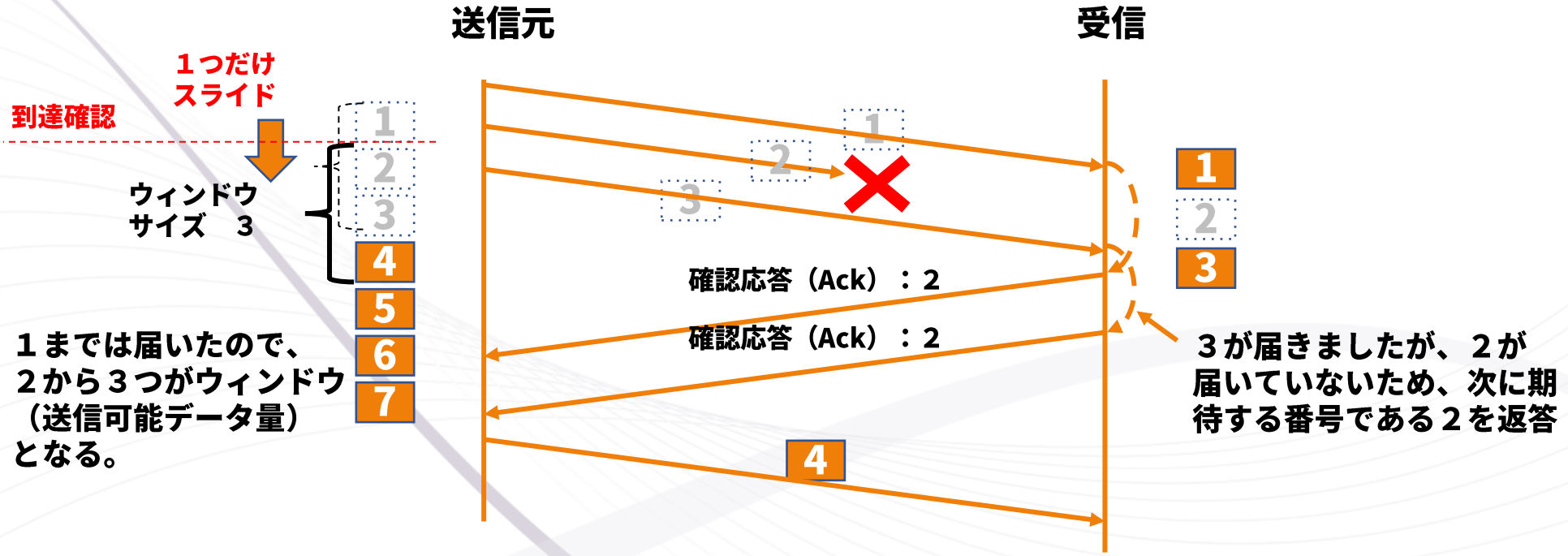
2-3.　ウィンドウ制御

【全て正常に届いた場合】



2-3.　ウィンドウ制御

【パケット２が損失した場合】



2-4.　ウィンドウサイズの決定

ウィンドウサイズは送信できるサイズなので送信端末で決定されます。
決定には以下の２つの内容の小さい値で決定されます。

①フロー制御

受信端末から通知されるバッファサイズ（ウィンドウサイズ）により制御

②輻輳制御

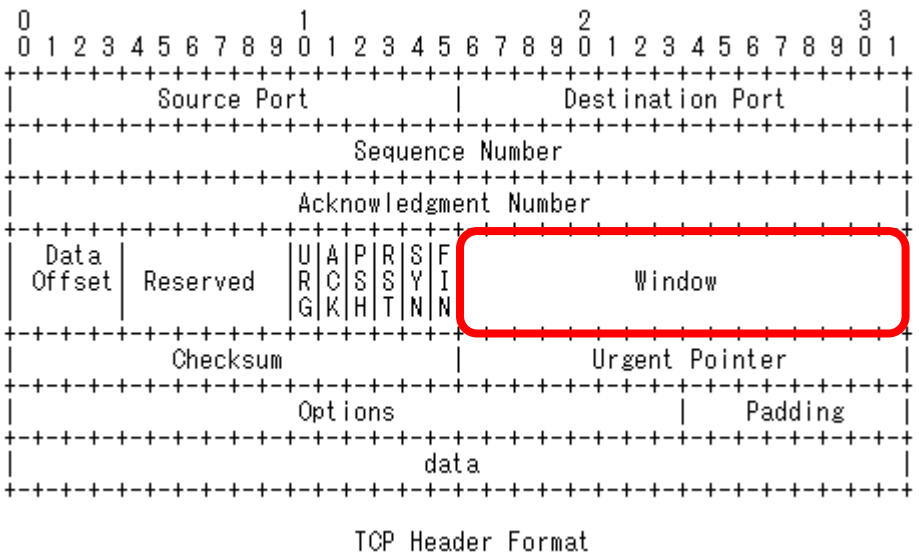
送信端末で計算されるウィンドウサイズにより制御

現在はパソコンの性能も向上しているため、受信端末のバッファサイズが影響することは少なく、送信側で計算される輻輳制御がウィンドウサイズになると考えて良いかと思います。

2-4-1. フロー制御

①に関しては、受信端末のバッファメモリが通知

- TCPパケットヘッダ「Window（16bit）」で通知
- 最大サイズ16bitで表示できる65535Byte（64Kbyte）
- 近年は最大サイズを拡張できるウィンドウサイズオプションの利用が一般的



TCP Header Format

2-4-2.　輻輳制御（輻輳ウィンドウ）

②輻輳制御は輻輳ウィンドウによって制御される。

輻輳ウィンドウは送信元端末の内部で保持し計算される値

パケットヘッダ内には情報は出てこない。

パソコンの高性能化により①フロー制御の受信端末バッファサイズよりも②輻輳制御の送信元端末で計算される輻輳ウィンドウサイズがウィンドウサイズの支配的。

（ただし、オプションなしでは16Bitの64Kbyteが最大値となり支配的であるため、ウィンドウサイズオプションにより拡張されるのが一般的です。）

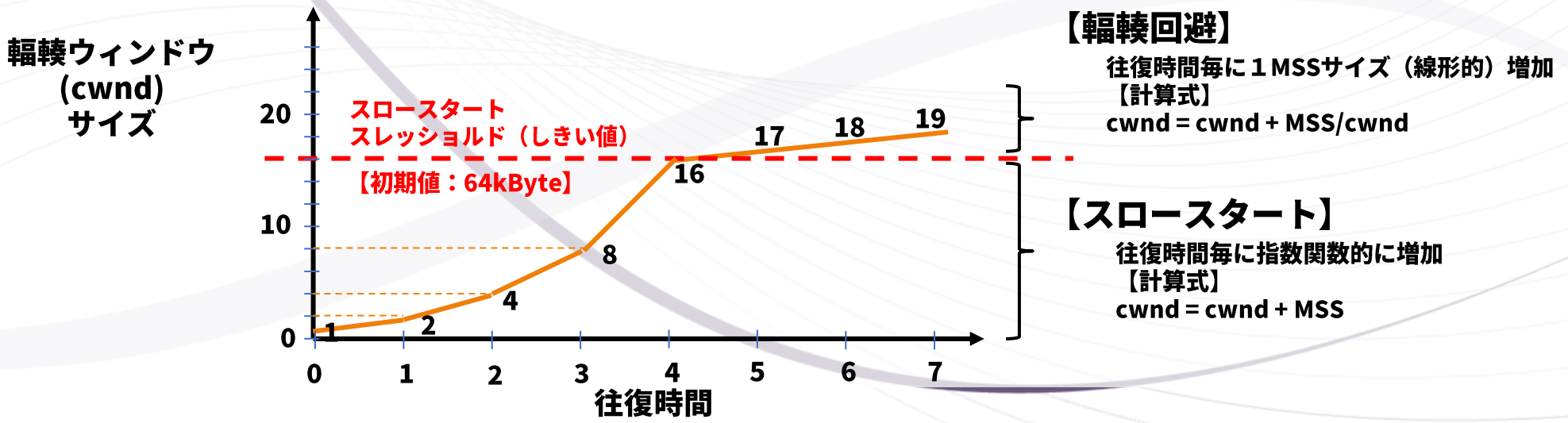
2-4-2.　輻輳制御（輻輳ウィンドウ）

輻輳ウィンドウ(cwnd: Congestion Window)：Byteサイズ（説明はパケット数）

輻輳ウィンドウサイズはMSS（MaximumSegmentSize）単位で計算されます。

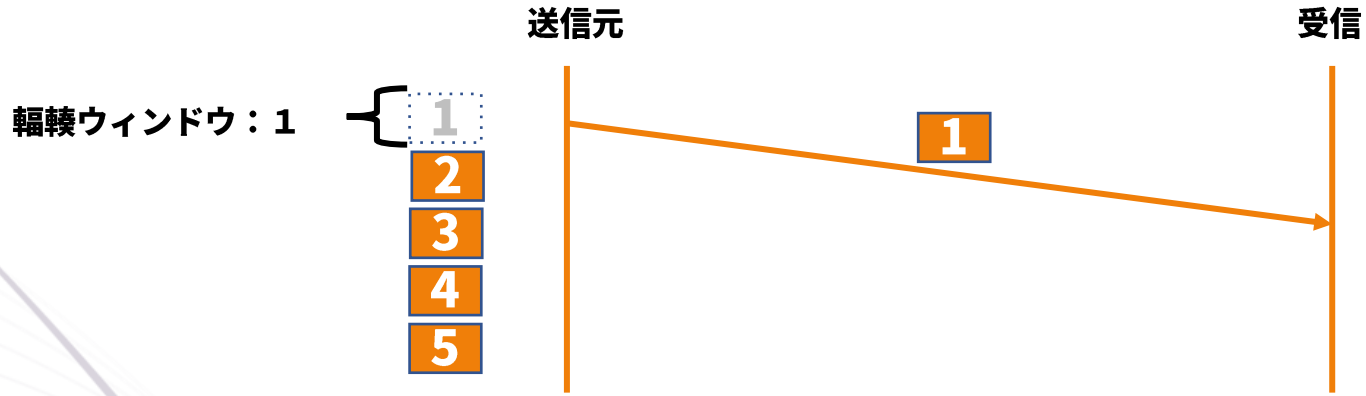
MSS　＝　Ether最大パケットサイズ　－　IPヘッダ長　－　TCPヘッダ長
　　＝　　1500 Byte　　－　　20 Byte　　－　　20 Byte（オプション無）
　　＝　1460 Byte

【スロースタート】と【輻輳回避】の２つのフェーズがあり、輻輳ウィンドウサイズの算出方法が異なる。

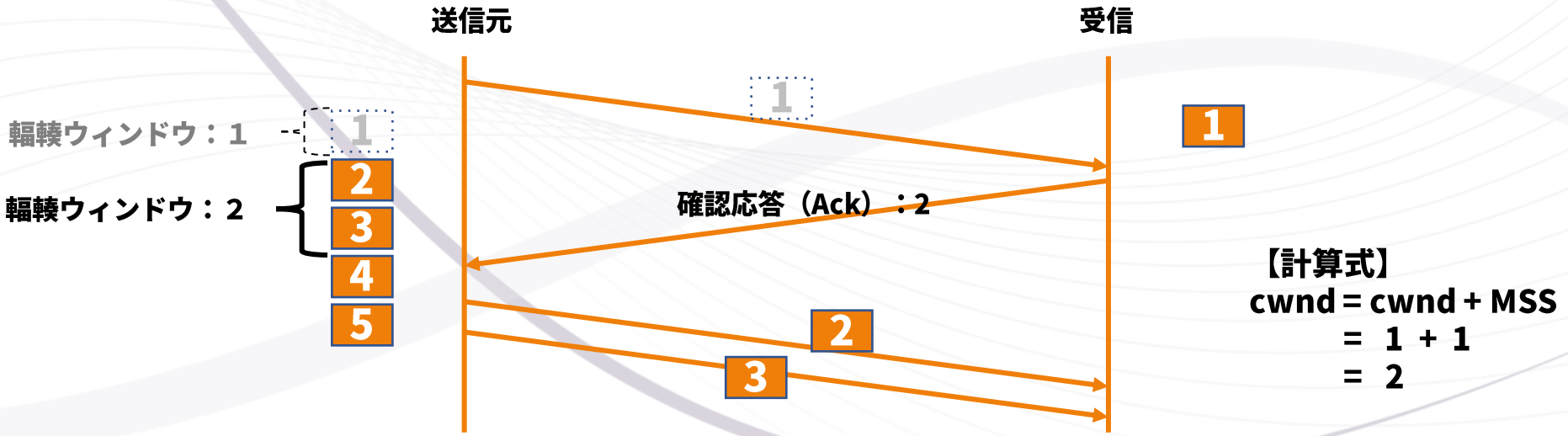


2-4-2-1.　スロースタート

【コネクション確立時】

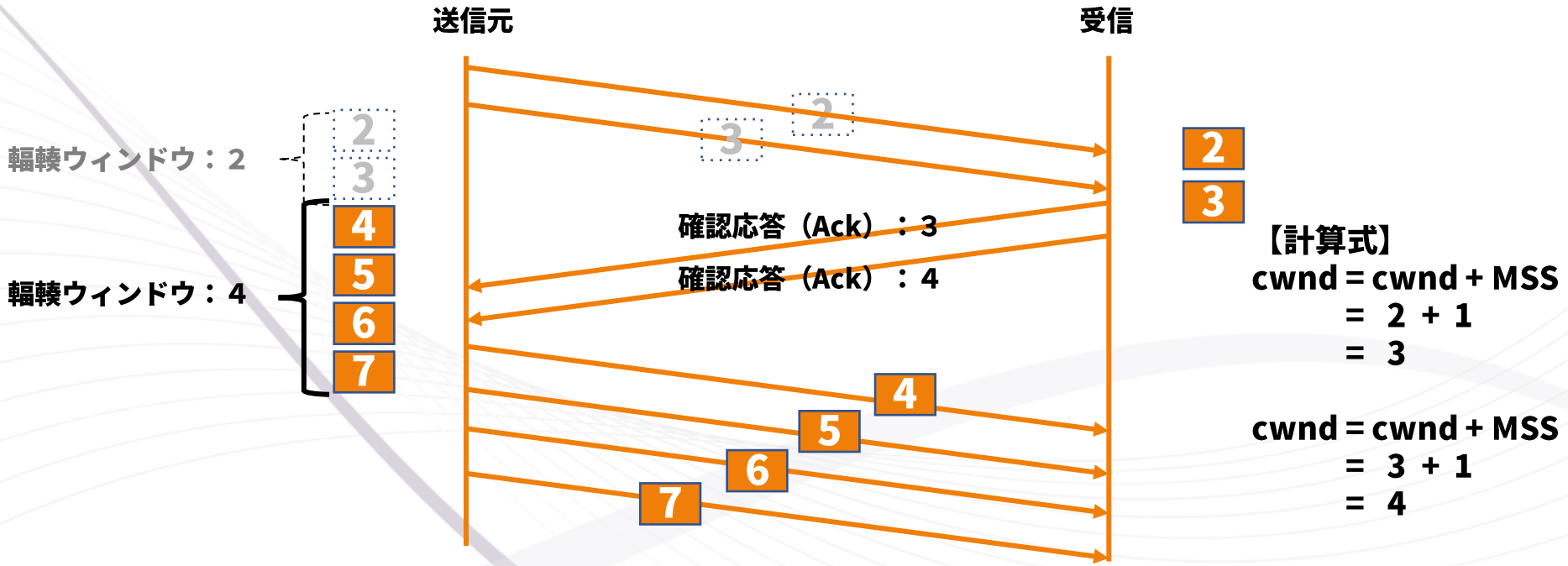


【確認応答１受信時】



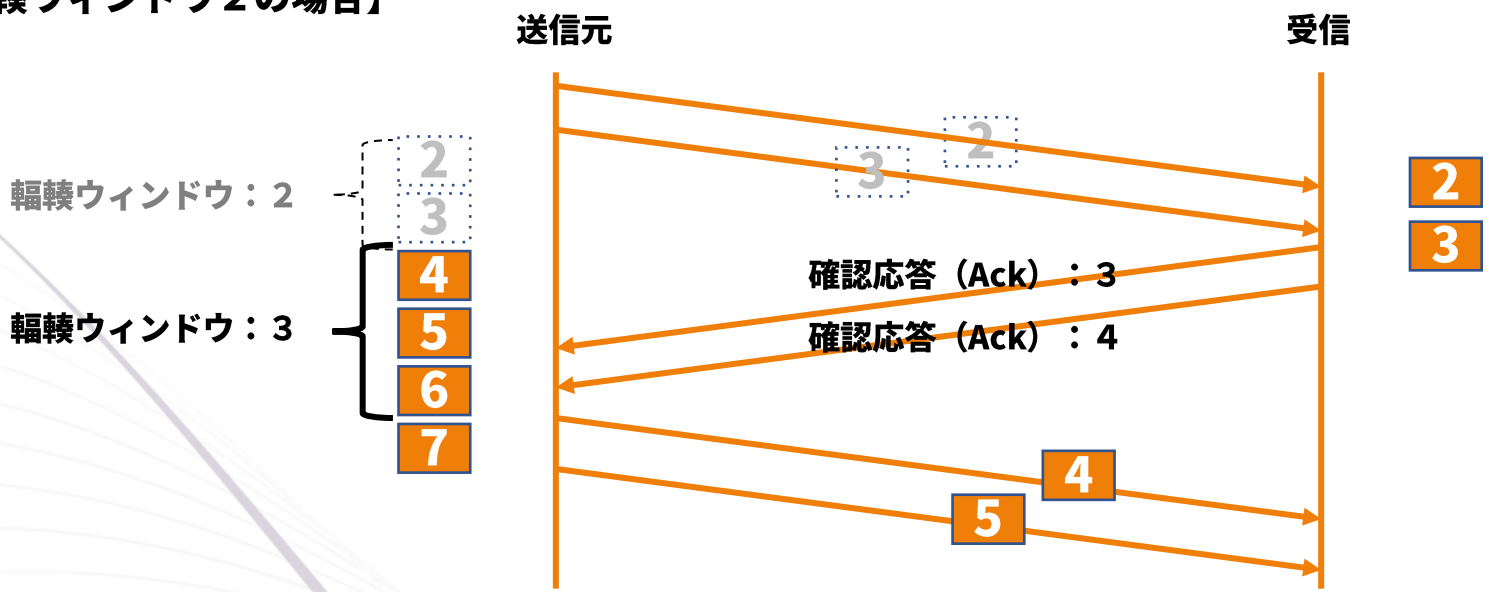
2-4-2-1.　スロースタート

【確認応答３まで受信時】



2-4-2-2.　輻輳回避

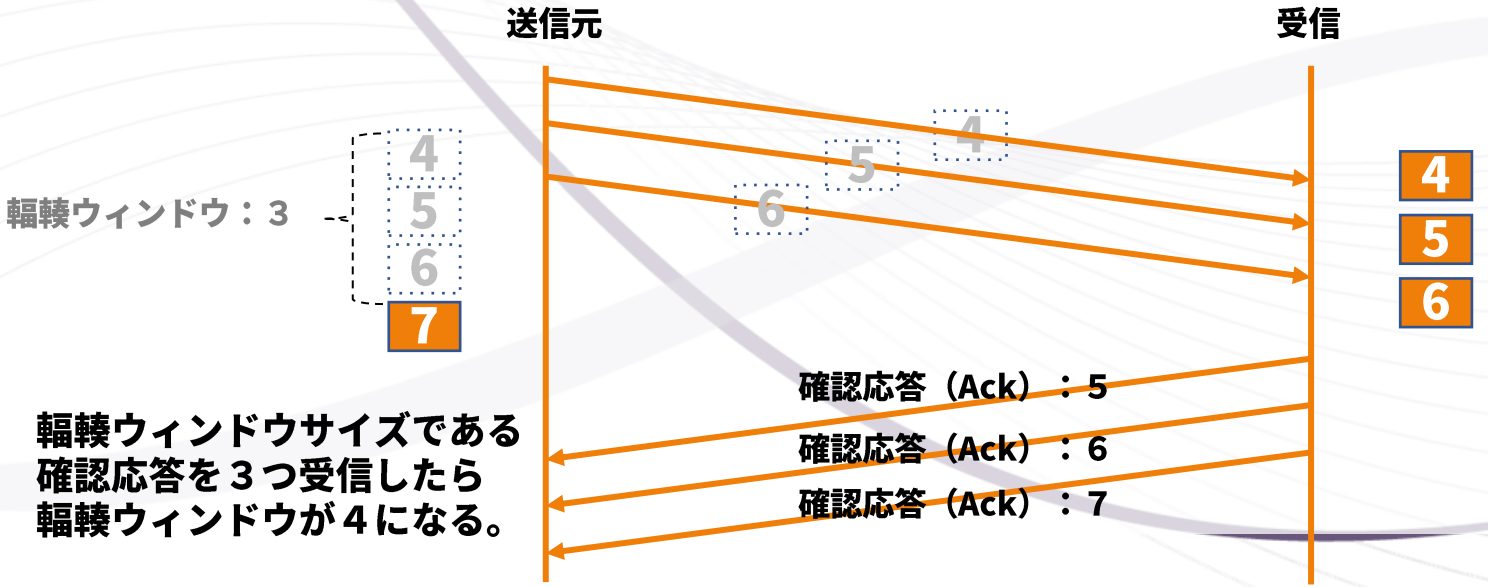
【輻輳ウィンドウ２の場合】



【計算式】
cwnd＝cwnd＋MSS/cwnd
＝　２　＋1/2
＝　２1/2

cwnd＝cwnd＋MSS/cwnd
＝　２1/2　＋1/2
＝　３

【輻輳ウィンドウ３の場合】



輻輳ウィンドウサイズである
確認応答を３つ受信したら
輻輳ウィンドウが４になる。

【計算式】
cwnd＝cwnd＋MSS/cwnd
＝　３　＋1/3
＝　３1/3

cwnd＝cwnd＋MSS/cwnd
＝　３1/3　＋1/3
＝　３2/3

cwnd＝cwnd＋MSS/cwnd
＝　３2/3＋1/3
＝　４

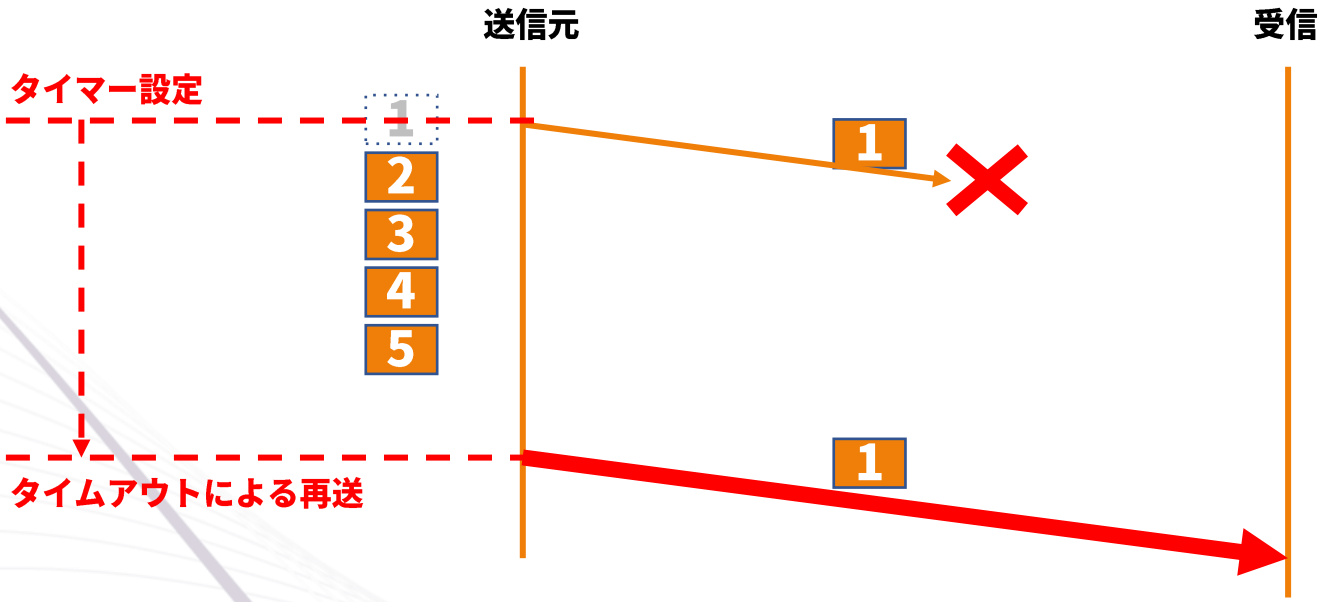
2-5.　再送制御

インターネットでは、ネットワーク内でパケットが損失することが前提で構築されています。
そのため、TCPでは確実にデータを届けるため再送制御機能を有しています。
再送機能については以下の２点の機能を実装しています。

①タイムアウトによる再送

②重複Ackによる再送

2-5-1.　タイムアウトによる再送

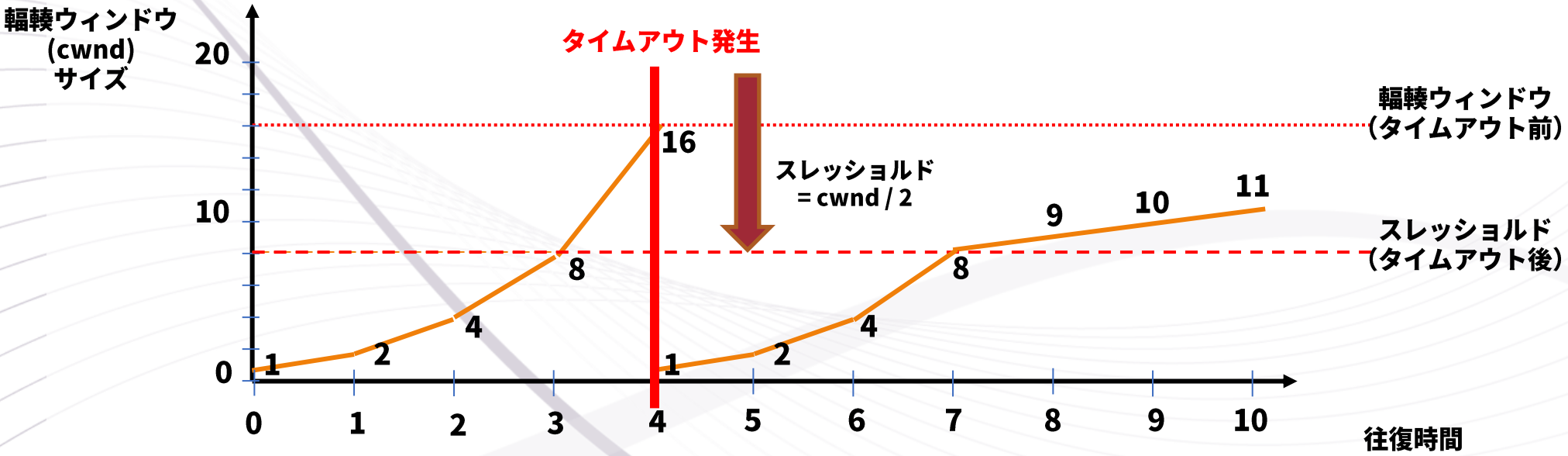


タイマー時間は初期値は決まっていますが、実際のパケットの応答時間から随時更新されます。

2-5-1. タイムアウトによる再送

タイムアウトによる再送が発生した場合には、以下のウィンドウ制御が行われ、輻輳ウィンドウサイズの変動を図にすると以下のようになります。

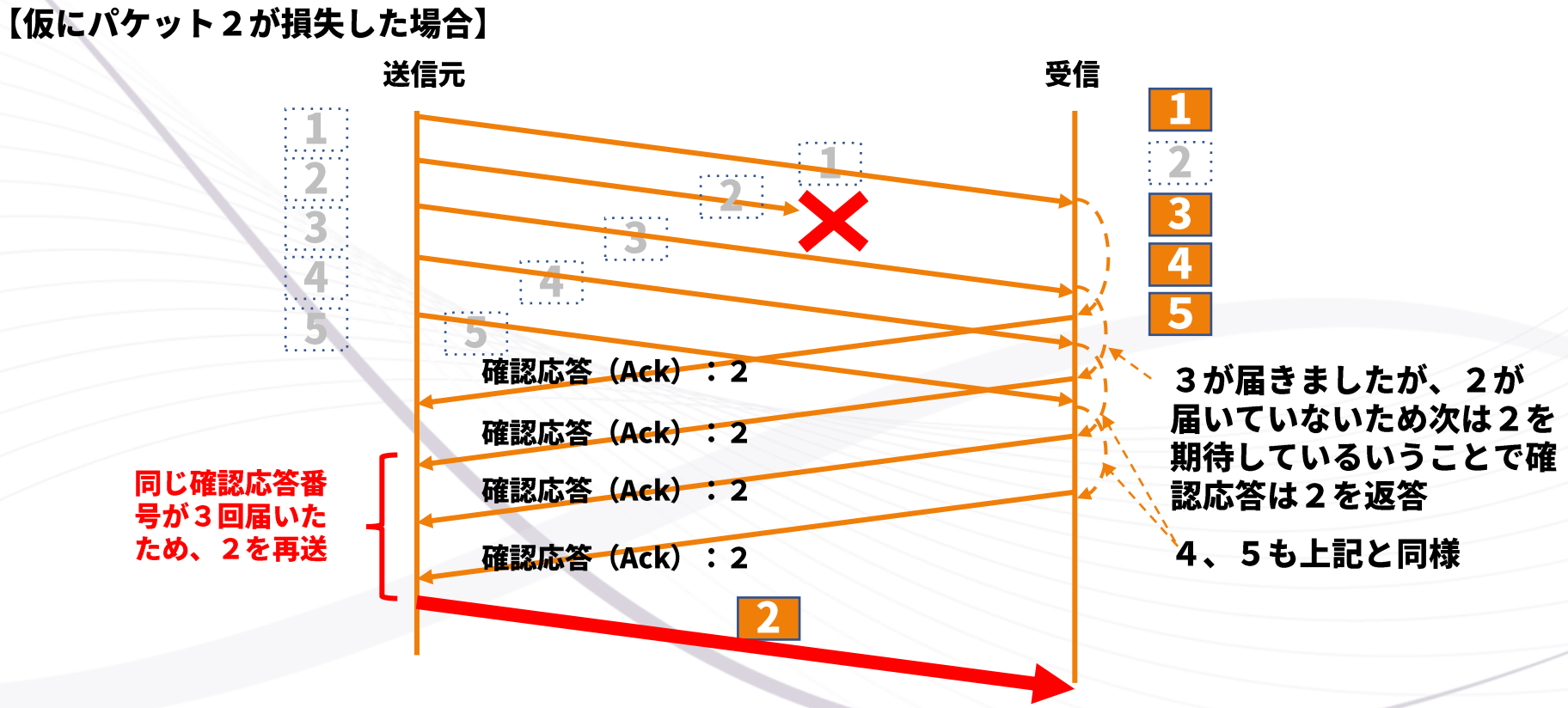
①輻輳ウィンドウ(cwnd)　　＝　１MSSサイズ
②スロースタートスレッショルド　＝　cwnd / 2



タイムアウトするということはネットワークは非常に混雑していると判断します。そのため、無駄なパケット損失を招かないように慎重に送信して少しづつウィンドウを広げていく必要があります。

2-5-2.　重複Ackによる再送

「高速再転送アルゴリズム」
重複Ackによる再送は、以前と同じシーケンス番号が３回届いたときに再送。

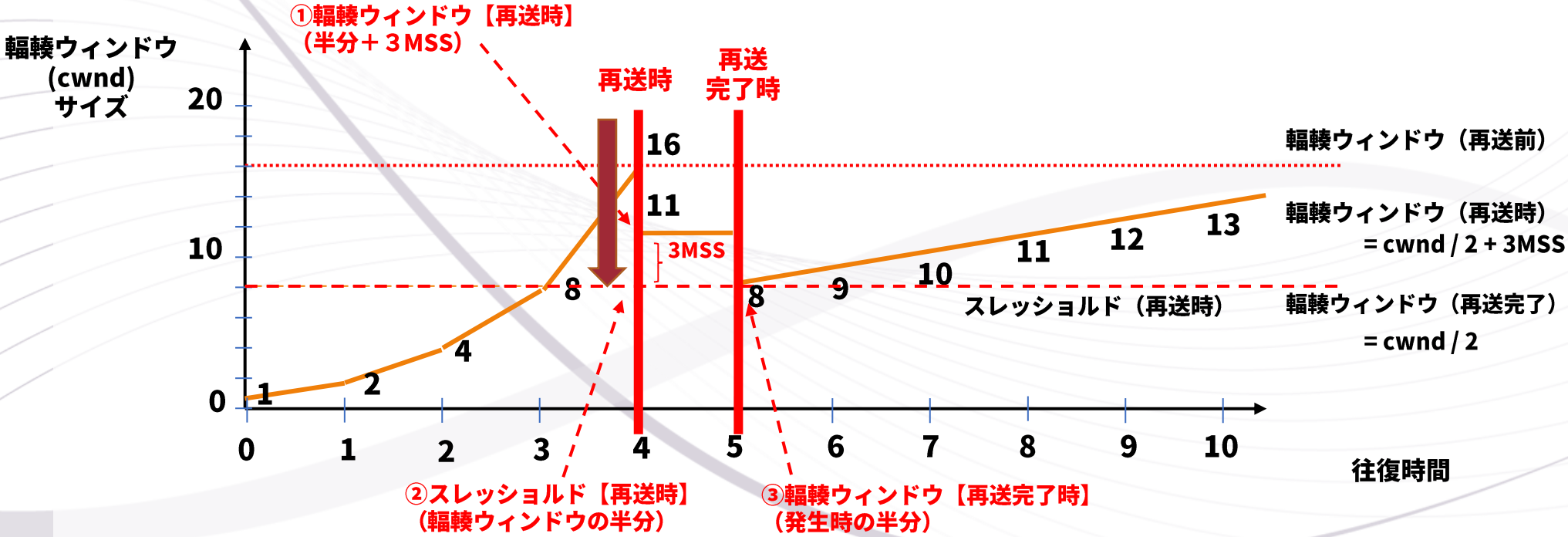


2-5-2.　重複Ackによる再送

高速リカバリ（FastRecorvery）機能：輻輳ウィンドウを最初の１MSSサイズではなく、以下の計算を実装し、再送によるデータ伝送を効率的に行っています。

【再送発生時】
①輻輳ウィンドウ(cwnd)　　＝　cwnd / 2　＋　３MSS
②スロースタートスレッショルド　＝　cwnd / 2

【再送完了時】
③輻輳ウィンドウ(cwnd)　　＝　スロースタートスレッショルド



3.　まとめ【TCPの主な伝送制御】

●到達確認

●ウィンドウ制御
- フロー制御
- 輻輳制御
  - -スロースタート
  - -輻輳回避

●再送制御
- タイムアウトによる再送　←【混雑大】
- 重複Ａｃｋによる再送　←【混雑小】

（高速再転送アルゴリズム）

チャンネル登録、ご意見、ご要望の程

よろしくお願いします。